(51) International Patent Classification[7]: H04L 12/46

(21) International Application Number:
PCT/US2003/030957

(22) International Filing Date: 1 October 2003 (01.10.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/322,120   17 December 2002 (17.12.2002)   US

(71) Applicant: RAYTHEON COMPANY [US/US]; 870 Winter Street, Waltham, MA 02451 (US).

(72) Inventors: ROSENGARD, Phillip, I.; 20986 Flatboat Court, Sterling, VA 20165-7443 (US). KRUNZ, Marwan, M.; 6601 East Ghost Flower Drive, Tucson, AZ 85750 (US).

(74) Agent: ICHIYE, Keiko; Baker Botts L.L.P., 2001 Ross Avenue, Suite 600, Dallas, TX 75201 (US).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:
— as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for all designations
— as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii)) for all designations

Published:
— with international search report

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND SYSTEM FOR ENCAPSULATING PACKETS
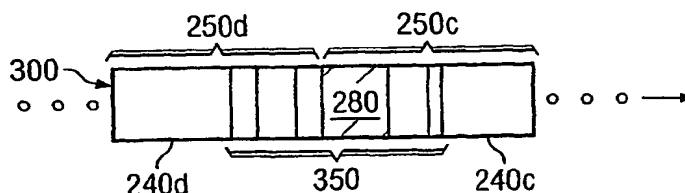
(57) Abstract: Encapsulating packets includes receiving packets at a queue of an encapsulator. The following operations are repeated until certain criteria are satisfied. A number of packets are accumulated at the queue. A current encapsulation efficiency associated with the accumulated packets is determined. A next encapsulation efficiency associated with the accumulated packets and a predicted next packet is determined. If the current encapsulation efficiency satisfies an encapsulation efficiency threshold and if the current encapsulation efficiency is greater than the next encapsulation efficiency, the accumulated packets are encapsulated. Otherwise, the packets continue to be accumulated at the queue.

METHOD AND SYSTEM FOR ENCAPSULATING PACKETS

TECHNICAL FIELD OF THE INVENTION

This invention relates generally to the field of data communication and more specifically to a method and system for encapsulating variable-size packets.

5   ## BACKGROUND OF THE INVENTION

Encapsulating packets in a communication system may involve the use of multiple queues for buffering packets waiting to be encapsulated. Packets at different queues, however, may experience different waiting times prior to encapsulation, also known as packet delay variation. Packet delay variation may introduce unwanted jitter into the

10   communication system. Moreover, encapsulation according to known techniques may result in sub-optimal bandwidth usage of a communications channel. Furthermore, packets associated with Internet Protocol (IP) traffic may involve datagrams of variable size, which may affect encapsulation for optimal bandwidth usage. Consequently, encapsulating packets while controlling jitter and enhancing bandwidth utilization has

15   posed challenges.

SUMMARY OF THE INVENTION

In accordance with the present invention, disadvantages and problems associated with previous techniques for encapsulation of variable-size packets in data communication

20   may be reduced or eliminated.

According to one embodiment, encapsulating packets includes receiving packets at a queue of an encapsulator. The following operations are repeated until certain criteria are satisfied. A number of packets are accumulated at the queue. A current encapsulation efficiency associated with the accumulated packets is determined. A next encapsulation

25   efficiency associated with the accumulated packets and a predicted size and arrival time of the next packet is determined. If the current encapsulation efficiency satisfies an encapsulation efficiency threshold and if the current encapsulation efficiency is greater than the next encapsulation efficiency, the accumulated packets are encapsulated.

2

Certain embodiments of the invention may provide one or more technical advantages. A technical advantage of one embodiment may be that the size of an encapsulation section is adjusted in response to a current encapsulation efficiency and a predicted jitter in an effort to control jitter while maintaining efficiency. If the
5   accumulated packets satisfy an encapsulation efficiency threshold, the packets may be encapsulated to maintain efficiency while controlling delay jitter. Another technical advantage of an embodiment may be that a delay associated with receiving predicted packets may be determined and if the delay satisfies a jitter requirement, the accumulated packets may be encapsulated to control jitter. Yet another technical advantage of an
10 embodiment may be that encapsulation sections packets may be formed into cells to improve jitter. The encapsulation sections may be formed into cells so that wasted, or additional, capacity in the cells is avoided.

Certain embodiments of the invention may include none, some, or all of the above technical advantages. One or more other technical advantages may be readily apparent to
15 one skilled in the art from the figures, descriptions, and claims included herein.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention and its features and advantages, reference is now made to the following description, taken in conjunction with
20 the accompanying drawings, in which:

FIGURE 1 illustrates an embodiment of a system for encapsulating packets to form encapsulation sections;

FIGURE 2 illustrates an embodiment of the formation of cells from encapsulation sections;

25 FIGURE 3 illustrates an embodiment of a cell flow;

FIGURE 4 is a flowchart illustrating a method for encapsulating packets;

FIGURE 5 illustrates another embodiment of the formation of cells from encapsulation sections; and

FIGURE 6 illustrates another embodiment of a cell flow.

3

DETAILED DESCRIPTION OF THE DRAWINGS

FIGURE 1 illustrates a system 10 for encapsulating packets to form encapsulation sections. System 10 adjusts the size of the sections in order to achieve a high encapsulation efficiency while ensuring satisfaction of a jitter requirement. In general, encapsulating a large section increases jitter. However, encapsulating a small section, while reducing jitter, results in lower efficiency. System 10 predicts jitter and efficiency, and adjusts the size of the encapsulation sections to reduce jitter while maintaining efficiency.

System 10 receives packets from real-time flows 15 and non-real-time flows 20, encapsulates the received packets to form encapsulation sections, forms the encapsulation sections into cells, and transmits the cells to a receiver 70. Packets may be associated with datagrams of different sizes or variable-size packets. For example, a packet may comprise an IP packet or datagram, which may be of any suitable size. Cells may generally be associated with fixed-size packet of data. For example, a cell may comprise a moving pictures experts group-2 packet (MPEG-2 packet). Any other suitable packet of fixed or variable size may be formed into cells without deviating from the concept of this invention.

Real-time flows 15 transmit real-time traffic, and non-real-time flows 20 transmit non-real-time traffic. According to one embodiment, real-time flows 15 may comprise any delay-sensitive traffic, for example, voice data such as voice over IP (VOIP) or video on demand (VOD) data. Non-real-time flows 20 may include any delay-tolerant traffic, for example, IP data. Flows that transmit other types of delay-sensitive traffic such as voice traffic or other real-time traffic may be used in place of or in addition to real-time flows 15.

System 10 may receive any suitable type of traffic, for example, moving pictures experts group-2 (MPEG-2) or MPEG-4 video traffic, voice over Internet protocol (VOIP), or Internet protocol (IP) packet traffic. The traffic may be classified according to jitter tolerance. According to one embodiment, traffic that is jitter tolerant comprises non-real-time traffic, and traffic that is not jitter tolerant comprises real-time traffic. Jitter tolerant traffic, however, may comprise any traffic that is jitter tolerant according to any suitable definition of "jitter tolerant," and jitter intolerant traffic may comprise any traffic that is not jitter tolerant. For example, jitter intolerant traffic may include voice traffic.

4

        System 10 includes real-time per-flow queues 40, non-real-time per-flow queues 30, and a processor 80. Real-time per-flow queues 40 buffer real-time traffic from real-time flows 15, and non-real-time per-flow queues 30 buffer non-real-time traffic from non-real-time flows 20. Each real-time per-flow queue 40a may receive from real-time

5   flows 15 a real-time flow 15a corresponding to the real-time per-flow queue 40a. Similarly, each non-real-time per-flow queue 30a receives a non-real-time flow 20a associated with the non-real-time per-flow queue 30a. As used in this document, "each" refers to each member of a set or each member of a subset of the set. Any number n_v of real-time flows 15 may be used and any suitable number n_d of non-real-time flows 20 may

10   be used. Similarly, any suitable number of real-time per-flow queues $40n_v$ and non-real-time per-flow queues $30n_d$ may be used. According to one embodiment, queues that queue other types of traffic such as voice traffic or other real-time traffic may be used in place of or in addition to real-time per-flow queues 40 or non-real-time per-flow queues 30.

15   Processor 80 manages the encapsulation process that starts when an incoming packet is received at a real-time per-flow queue 40. Processor 80 determines if the packets accumulated at the real-time per-flow queue 40 form an encapsulation section of a maximum section size. If the packets do not form an encapsulation section of a maximum section size, processor 80 determines if a current encapsulation efficiency of the

20   accumulated packets satisfies an encapsulation efficiency threshold. Processor 80 may predict a next packet size to determine if the current encapsulation efficiency would improve by waiting for the next packet. Processor 80 may also predict a next packet arrival time to determine whether a jitter requirement would be violated by waiting for the next packet. The processor 80 may adjust the size of sections to be encapsulated at the

25   corresponding real-time per-flow queues 40, and may also adjust the number of cells formed from the encapsulated sections. In addition, processor 80 may adjust the number of packets received from non-real-time per-flow queues 30 that are encapsulated to form sections and cells.

        Encapsulation sections from queues 40 and 30 are formed into cells. Cells of real-

30   time encapsulation sections are copied into common real-time buffer 50, and cells of non-real-time encapsulation sections are copied into common non-real time buffer 55. The cells of an encapsulation section may be copied sequentially into real-time buffer 50 or

5

non-real time buffer 55 such that the cells are not interleaved by cells of another encapsulation section. Real-time buffer 50 and non-real time buffer 55 may process the cells according to a first-in-first-out procedure. According to one embodiment, buffers that buffer other types of traffic such as voice traffic or other real-time traffic may be used
5   in place of or in addition to real-time buffer 50 or non-real time buffer 55.

System 10 may also include a scheduler 60 that outputs real-time cells from real-time buffer 50 and non-real-time cells from non-real time buffer 55. Cells received at the real-time buffer 50 and non-real time buffer 55 may be handled by the scheduler 60 in order of priority. For example, real-time cells associated with the real-time buffer 50 may
10   be assigned higher priority than non-real-time cells associated with the non-real time buffer 55 such that non-real-time cells from non-real time buffer 55 are transmitted only if real-time buffer 50 is empty. Accordingly, non-real time buffer 55 may be sufficiently large to store delayed non-real-time encapsulation sections. The scheduler 60 may further transmit the cells from the real-time buffer 50 and non-real time buffer 55 to a receiver 70.
15   Any other suitable priority-scheduling schemes may be used to arbitrate between the real-time buffers and the non-real-time buffers.

To summarize, system 10 encapsulates packets to form encapsulation sections. The number of packets to be encapsulated at each real-time per-flow queue 40 is adjusted in response to an encapsulation efficiency threshold and a jitter requirement, with the goal
20   of reducing jitter while maintaining efficiency. Encapsulation sections are formed into cells. One embodiment of the formation of cells from encapsulation sections is to be described with reference to FIGURE 2. A cell flow may be transmitted to receiver 70. One embodiment of a cell flow is described with reference to FIGURE 3. A method for determining when to encapsulate packets is described with reference to FIGURE 4.
25   Another embodiment of the formation of cells from encapsulation sections is described with reference to FIGURE 5. Another embodiment of a cell flow is described with reference to FIGURE 6.

FIGURE 2 illustrates an embodiment of the formation of cells 250 from encapsulation section 200. According to one embodiment, encapsulation section 200 may
30   comprise a Multi-Protocol Encapsulation (MPE) section. Encapsulation section 200 includes a section header 210, section data 212, and a section footer 214. Section header 210 may include, for example, digital video broadcasting (DVB) Multi-Protocol

6

Encapsulation (MPE) header data. Section header 210 may also include, for example, Digital Storage Media Command and Control (DSM-CC) header data. Section data 212 includes packets 216. According to one embodiment, each packet 216 may comprise an IP packet or datagram. Section footer 214 may include, for example, error correction
5   codes. According to one embodiment, datagram sizes are variable, while section header 210 has, for example, 20 bytes, and section footer 214 has, for example, 4 bytes.

According to one embodiment, encapsulation section 200 may be formed into cells 250. Each cell 250 includes a cell header 220, a payload 240, and a cell footer 230. Cell header 220 may include, for example, MPEG-2 packet header data. Cell footer 230 may
10   include, for example, MPEG-2 packet footer data. Payload 240 may include MPEG-2 packets which have, according to one embodiment, 184 bytes.

According to one embodiment, an encapsulation section 200 may be formed into one or more sequential cells 250. In the illustrated example, a cell 250a may be loaded with a first portion of section 200, a next cell 250b may be loaded with a continuing
15   portion of section 200, and a cell 250c may be loaded with a last portion of section 200. In general, cells 250 may be formed sequentially until the encapsulation section 200 has been fragmented into one or more cells 250. According to one embodiment, a cell 250 loaded with a last portion of an encapsulation section 200 may include additional capacity 280. In the illustrated example, a last portion 245 of encapsulation section 200, comprising a
20   section data 216c and section footers 214, is loaded into the payload of cell 250c to form a loaded portion of the payload 240c. The last portion 245 does not fill the capacity of the payload, so cell 250c has additional capacity 280. However, cells 250 may be formed in any other suitable manner. For example, another embodiment of forming cells 250 is described with reference to FIGURES 5 and 6.

25   FIGURE 3 illustrates an embodiment of a cell flow 300 resulting from the formation of cells 250 as described with reference to FIGURE 2. In general, the cell flow 300 may include cells 250 ordered sequentially, consecutively, successively, serially, or in any similar manner therewith. According to one embodiment, each cell 250 may include, for example, IP cells or datagrams 240. In the illustrated example, a last portion of cell
30   250c associated with datagram 240c may be followed by a first portion of a cell 250d associated with datagram 240d. According to one embodiment, a cell 250c may include additional capacity 280. Additional capacity 280 may result from the formation of cells as

illustrated in FIGURE 2. In the illustrated example, a cell flow 300 containing additional capacity 280 may contribute to a delay 350. Delay 350 or jitter experienced in a cell flow 300 may be calculated as the time lapsed between a last IP datagram 240c and next IP datagram 240d. Cell flow 300 may be formed in any other suitable manner, for example, with the exclusion of additional capacity 280 as described with reference to FIGURES 5 and 6.

FIGURE 4 is a flowchart illustrating an embodiment of a method for encapsulating IP packets. The following parameters may be used to perform calculations described in the examples illustrated with reference to FIGURE 4.

Parameter	Definition
n	Total number of real-time and non-real-time flows.
n_v	Number of real-time flows
n_d	Number of non-real-time flows
$\eta(\chi)$	Encapsulation efficiency for a section with an χ -byte payload
η_{opt}	Optimal encapsulation efficiency
η_{rt}	Threshold that defines the minimum acceptable encapsulation efficiency for real-time traffic
η_{nrt}	Threshold that defines the minimum acceptable encapsulation efficiency for non-real time traffic
S_{opt}	Optimal payload size for a section
$T_i(j)$	jth cell inter-arrival time in the ith flow
$S_i(j)$	Size (in bytes) of the jth cell in the ith flow
$\mu_{inter}(i,j)$	Exponentially weighted moving average of $T_i(j)$
$\mu_{size}(i,j)$	Exponentially weighted moving average of $S_i(j)$
$\tau_i(j)$	$T_i(j) - \mu_{inter}(i,j)$
$s_i(j)$	$S_i(j) - \mu_{size}(i,j)$
ff_{inter}	Forgetting factor for updating $\mu_{inter}(i,j)$
ff_{size}	Forgetting factor for updating $\mu_{size}(i,j)$
$T_i(m;r)$	r-step forecast for the inter-arrival time of the ith flow, starting form the mth inter-arrival time
$S_i(m;r)$	r-step forecast for the cell size of the ith flow, starting from the mth cell size
$\alpha_i(m)$	Coefficient of the autoregressive(1) model for the cell inter-arrival time in the ith flow after the arrival of the mth cell
$\hat{\alpha}_i(m)$	Estimated value of $\alpha_i(m)$
$\beta_i(m)$	Coefficient of the autoregressive(1) model for the cell size in the ith flow after the arrival of the mth cell
$\hat{\beta}_i(m)$	Estimated value of $\beta_i(m)$
Q_{max}	Maximum number of bytes of an encapsulation section
Q_{RT}	Number of MPEG2 packets in the real-time buffer at the current

	time
$Q_{RT}^{pred}(t)$	Predicted number of Q_{RT} after t seconds
d_i	Worst case jitter experienced by cells of the ith flow
$\hat{d}_i$	Predicted value of d_i after one (future) inter-arrival time
D_i	Jitter requirement for the ith flow
R_{encap}	Output rate of the encapsulator in bps
R_i	Peak input rate for the ith flow in bps
$S^{(i)}_{tot}$	Number of bytes in the ith per-flow queue at a given time instant
$S^{(i)}_{tot}(x)$	Predicted value of $S^{(i)}_{tot}$ after x seconds
$S^{(i)}_{current}$	Size of the most recent cell to arrive from the ith flow in seconds
$\hat{S}^{(i)}_{next}$	Predicted size of the subsequent inter-arrival interval in the ith flow in seconds
$T^{(i)}_{current}$	Duration of the most recent inter-arrival time in the ith flow (in seconds)
$\hat{T}^{(i)}_{current}$	Predicted length of the subsequent inter-arrival interval in the ith flow (in seconds)
$t^{(i)}_{start}(k)$	Arrival time of the first bit of the kth cell in the ith flow
$t^{(i)}_{start,1}$	Arrival time for the first bit of the last cell of an MPE section in the ith flow
$t^{(i)}_{start,2}$	Arrival time for the first bit of the first cell of an MPE section in the ith flow
$t^{(i)}_{finish,1}$	Departure time for the first bit of the last cell of an MPE section in the ith flow
$t^{(i)}_{finish,2}$	Departure time for the first bit of the first cell of an MPE section in the ith flow
$\hat{t}^{(i)}_{finish,2}$	Predicted value of $t^{(i)}_{finish,2}$
$W^{(i)}$	$\hat{T}^{(i)}_{next} + 8\hat{S}^{(i)}_{next}/R_i$
$t_{current}$	Current time at the encapsulator
$m^{(j)}_{last}$	Index of the most recent cell to arrive at the jth queue

In the illustrated method, packets are received at the per-flow queues 40 and 30 from real-time flows 15 and non-real-time flows 20. As an example, a packet of real-time flow 15a is received by real-time per-flow queue 40a. In describing the embodiment of FIGURE 4, the term "per-flow queue" refers to a real-time per-flow queue 40 or a non-real-time per-flow queue 30.

The method begins at step 400, where a per-flow queue receives a packet. The method then proceeds to step 402, where the size of an encapsulation section including the received packet is compared to a maximum number of bytes Q_{max} of an encapsulation section. The size of an encapsulation section describes the combined sizes of the one or

9

more packets stored at the per-flow queue. According to one embodiment, the maximum number of bytes Q_{max} may be 4080 bytes. If the size of the encapsulation section exceeds the maximum number of bytes Q_{max} associated with the per-flow queue, the method proceeds to step 416 to encapsulate the packet.

5   If the size of the encapsulation section does not satisfy the maximum number of bytes Q_{max} associated with the per-flow queue, the method proceeds to step 404 to determine if a current encapsulation efficiency $\eta(\chi)$ of the one or more packets accumulated at the per-flow queue satisfies a first threshold. In general, step 404 determines if a current encapsulation efficiency $\eta(\chi)$ satisfies a real-time encapsulation

10 efficiency threshold η_{rt} or a non-real time encapsulation efficiency threshold η_{nrt}. According to one embodiment, the current encapsulation efficiency may be defined as, for example, the ratio of a number χ of payload bytes associated with an encapsulation section comprising one or more packets accumulated in the per-flow queue to the total number of bytes of the encapsulation section. According to one embodiment, the payload

15 bytes refer to the IP-traffic bytes that generate an encapsulation section. Accordingly, the current encapsulation efficiency may be described by Equation (1):

$$\eta(\chi) = \frac{\chi}{204\left\lceil\frac{(\chi+24)}{184}\right\rceil}, \text{ for } 1 <= \chi <= 4080 \tag{1}$$

20 where $\lceil . \rceil$ is the ceiling function. Referring to Equation (1), the larger the value of the number χ of payload bytes, the higher the current encapsulation efficiency $\eta(\chi)$. However, the trend may not be monotonic due to padding of unused bytes in a cell. For example, the optimal efficiency η_{opt}, for one embodiment, is achieved when number $\chi = S_{opt} = 4024$ bytes, as described by Equation (2):

25

$$\eta_{opt} = \max \eta(\chi) = \eta(4024) = 89.66\% \tag{2}$$

    According to one embodiment, real-time encapsulation efficiency threshold η_{rt} and non-real time encapsulation efficiency threshold η_{nrt} are predetermined encapsulation

30 efficiency thresholds associated with a corresponding real-time or non-real-time per-flow

queue, respectively. An encapsulation efficiency threshold may describe a minimum acceptable encapsulation efficiency. For example, the value of threshold η_{rt} may be set approximately at 80%. To summarize step 404, a current encapsulation efficiency $\eta(\chi)$ for packets received at the per-flow queue is obtained from Equation (1). The current
5 encapsulation efficiency $\eta(\chi)$ is compared with an encapsulation efficiency threshold to determine if the encapsulation efficiency threshold has been satisfied. If the current encapsulation efficiency does not satisfy the encapsulation efficiency threshold at step 404, the method proceeds to step 420 to wait for the next packet.

If the current encapsulation efficiency satisfies the encapsulation efficiency
10 threshold at step 404, the method proceeds to steps 406 and 408 to predict a next encapsulation efficiency and a next packet arrival time. According to one embodiment, a next encapsulation efficiency may be associated with one or more packets including a predicted next packet accumulated at a per-flow queue. In the illustrated example, predicting the next encapsulation efficiency comprises predicting a next packet size. The
15 next packet arrival time is calculated to predict delay.

According to one embodiment, steps 406 and 408 are calculated, for example, using a recursion procedure. According to another embodiment, predicting a next packet size and predicting a next packet arrival time may be obtained by an iterative procedure or any other procedure suitable to generate the predicted values of a next packet size and a
20 next packet arrival time.

For example, steps 406 and 408 may be performed in the following manner. A predicted next packet size $S_i(j)$, used to calculate a next encapsulation efficiency at step 406, may be associated with the size of bytes of the jth packet of the ith flow. Similarly, a predicted next packet arrival time $T_i(j)$ is associated with the jth packet arrival time of the
25 ith queue flow. The exponentially weighted moving averages (EMWAs) of $T_i(j)$ and $S_i(j)$ may be defined as $\mu_{inter}(i,j)$ and $\mu_{size}(i,j)$ given by Equations (3) and (4), respectively:

$$\mu_{inter}(i, j+1) = f\!f_{inter}\mu_{inter}(i,j) + (1 - f\!f_{inter})T_i(j+1) \qquad (3)$$

$$\mu_{size}(i, j+1) = f\!f_{size}\mu_{size}(i,j) + (1 - f\!f_{size})S_i(j+1) \qquad (4)$$

30

where ff_{inter} and ff_{size} are forgetting factors that determine the degree of adaptiveness in the estimation of the moving averages.

According to one embodiment, the next encapsulation efficiency calculated at step 406 and the prediction of next packet arrival time calculated at step 408, may include modeling a sequence according to an autoregressive model. In the illustrated example, an arrival time sequence may be modeled by defining $(\tau_i(j) \equiv T_i(j) - \mu_{inter}(i, j))$ and a packet size sequence may be modeled by defining $(s_i(j) \equiv S_i(j) - \mu_{size}(i,j))$, where $\tau_i(j)$ and $s_i(j)$ are normalized, zero-mean, packet inter-arrival times and sizes, respectively. Thus, according to one embodiment, for each i, the sequences $(\tau_i(j) : j=1,2,...)$ and $(s_i(j) : j=1,2,...)$ may be obtained according to an autoregressive model of order one (AR(1)) with time-varying coefficients as described by Equations (5) and (6):

$$\tau_i(k+1) = \alpha_i(k)\tau_i(k) + \sigma_i\varepsilon(k+1), k = 1,2,... \qquad (5)$$

$$s_i(k+1) = \beta_i(k)s_i(k) + \xi_i\varepsilon(k+1), k = 1,2,... \qquad (6)$$

where $\alpha_i(k)$ and $\beta_i(k)$ are time-varying coefficients, $\{\varepsilon(j) : j = 1,2,...\}$ is a sequence of independent identically distributed zero-mean random variables of the noise processes, and σ_i and ξ_i are variance-related coefficients of the noise processes. The modeling of a sequence may be performed using any autoregressive techniques.

As discussed previously, the predicted size of the next packet $S_i(j)$ and predicted next packet arrival time $T_i(j)$ are determined at steps 406 and 408, respectively. For an ith per-flow queue flow, where $i = 1, 2,..., n_v$, and j packets, where $j=1, 2,..., m$, and m is the index of the most recent packet to arrive at the encapsulator, the predicted values of $T_i(j)$ and $S_i(j)$ for $j > m$ during the r-step of a recursion procedure are described respectively by Equations (7) and (8):

$$\hat{T}_i(m;r) = (\hat{\alpha}_i(m))^r (T_i(m) - \mu_{inter}(i,m)) + \mu_{inter}(i,m) \qquad (7)$$

$$\hat{S}_i(m;r) = (\hat{\beta}_i(m))^r (S_i(m) - \mu_{size}(i,m)) + \mu_{size}(i,m) \qquad (8)$$

where the adjusted coefficients $\hat{\alpha}_i(m)$ and $\hat{\beta}_i(m)$ are obtained by minimizing the mean-square error as described by Equations (9) and (10):

12

$$\hat{\alpha}_i(m) = \frac{\sum_{j=1}^{m-1} T_i(j)T_i(j+1)}{\sum_{j=1}^{m-1} T_i^2(j)} \tag{9}$$

$$\hat{\beta}_i(m) = \frac{\sum_{j=1}^{m-1} S_i(j)S_i(j+1)}{\sum_{j=1}^{m-1} S_i^2(j)} \tag{10}$$

As previously described, the illustrated method may use a recursion procedure. According to one embodiment, the adjusted coefficient $\hat{\alpha}_i(m)$ may be generated recursively each time a next packet arrival time needs to be predicted. To predict a next packet arrival time called for at step 408, a value for $\hat{\alpha}_i(m)$ may be determined using recursive variables $Y_i(k)$ and $Z_i(k)$ as defined by Equations (11) and (12):

$$Y_i(k) \overset{def}{=} \sum_{j=1}^{k} T_i(j)T_i(j+1) = Y_i(k-1) + T_i(k)T_i(k+1) \tag{11}$$

$$Z_i(k) \overset{def}{=} \sum_{j=1}^{k} T_i^2(j) = Z_i(k-1) + T_i^2(k) \tag{12}$$

where the r-step prediction may be obtained using the following procedure once the mth measurement for the last arrival time is available:

For all $i = 1, \ldots, n_v$, do
    (Initialization) After the arrival of the first packet, do
        $\hat{T}_i(1;r) = \mu_{inter}(i,1) = T_i(1)$
        $Y_i = 0$
        $Z_i = 0$
    (Recursion) For each subsequent cell arrival, m, do
        Update EMWA: $\mu_{inter}(i,m) = ff_{inter}\mu_{inter}(i,m-1) + (1 - ff_{inter})T_i(m)$
        $Y_i = Y_i + T_i(m-1)T_i(m)$
        $Zi = Zi + T_i^2(m-1)$
        $\hat{\alpha}_i(m) = Y_i / Z_i$
        $\hat{T}_i(m;r) = (\hat{\alpha}_i(m))^r (T_i(m) - \mu_{inter}(i,m)) + \mu_{inter}(i,m)$
    End-recursion
End-for

The prediction of the next encapsulation efficiency at step 406 may be determined using a similar recursion procedure. Once the prediction of next encapsulation efficiency is determined, the method proceeds to step 410 to determine if the predicted next

5   encapsulation efficiency exceeds the current encapsulation efficiency previously determined at step 404 by Equation (1). If the next encapsulation efficiency is lower than the current encapsulation efficiency, the method proceeds to step 416 to encapsulate the packets accumulated at the per-flow queue.

If the next encapsulation efficiency is higher than the current encapsulation

10   efficiency, the method proceeds to step 412 to determine the expected delay associated with receiving the next packet. In the illustrated method, the expected delay may be defined as the absolute value of the difference between the delay of the first packet at the encapsulator and the delay of the next packet. According to one embodiment, when a first packet arrives at the encapsulator, the time of arrival of the first packet is noted until the

15   value is updated with the time of arrival of a subsequent or next packet. The time of arrival of the first bit of the kth packet in the ith flow queue may be represented by $t_{start}^{(i)}(k)$, where $k=1,2,...$, and $i=1,2,..., n_v$. The times of arrival and departure for the first bit of a first packet may be represented by $t_{start,1}^{(i)}$ and $t_{finish,1}^{(i)}$. Similarly, the times of arrival and departure for the first bit of a second packet may be represented by $t_{start,2}^{(i)}$ and

20   $t_{finish,2}^{(i)}$. The expected delay d_i may be defined by Equation (13):

$$d_i \stackrel{def}{=} \max\{0, (t_{finish,2}^{(i)} - t_{finish,1}^{(i)}) - (t_{start,2}^{(i)} - t_{start,1}^{(i)})\} \qquad (13)$$

In the worst case, the computation of d_i may be performed after each packet arrives at the

25   per-flow queue.

To initialize the process of determining delay d_i at step 412, $t_{start,1}^{(i)} = t_{finish,1}^{(i)} = 0$ and $t_{start,2}^{(i)} = t_{start,1}^{(i)}$. Subsequently, when an encapsulation section is formed, the value of $t_{start,1}^{(i)}$ is updated and used to update $t_{finish,1}^{(i)}$ as described by Equation (14):

$$t_{finish,1}^{(i)} = t_{start,1}^{(i)} + \frac{8S_{current}^{(i)}}{R_i} + \frac{Q_{RT} + \left\lceil \frac{S_{tot}^{(i)} - S_{current}^{(i)} + 20}{184} \right\rceil}{\frac{R_{encap}}{(204)(8)}} \qquad (14)$$

14

where $S_{tot}^{(i)}$ is the current number of bytes in the per-flow queue, $S_{current}^{(i)}$ is the size of the most recent packet to arrive from the ith per-flow queue, R_i is the peak rate in bits per second of the ith flow, R_{encap} is the rate of the encapsulator in bps, and Q_{RT} is the number of encapsulated cells in the real-time per-flow queue 40 at the time of computing $t_{finish,1}^{(i)}$.

According to one embodiment, time $t_{finish,1}^{(i)}$ may be generated by, for example, using expression $t_{start,1}^{(1)} + (8S_{current}^{(i)}/R_i)$, if it is computed at or after the last bit of the first packet arrives. After the encapsulation section including the first packet is formed, the value of time $t_{start,2}^{(i)}$ is updated.

Determining the delay d_i of Equation (13) at step 412 may require the prediction of time $t_{finish,2}^{(i)}$. The predicted value of time $t_{finish,2}^{(i)}$ may be referred to as $\hat{t}_{finish,2}^{(i)}$ and described by Equation (15):

$$\hat{t}_{finish,2}^{(i)} \approx t_{current} + W^{(i)} + \frac{Q_{RT}^{pred}(W^{(i)})(204)(8)}{R_{encap}} \tag{15}$$

where:

$$W^{(i)} \stackrel{def}{=} \hat{T}_{next}^{(i)} + \frac{8\hat{S}_{next}^{(i)}}{R_i} \tag{16}$$

and $Q_{RT}^{pred}(t)$ is the predicted number of cells at the real-time buffer 50 after t seconds from the current time $t_{current} (Q_{RT}^{pred}(0) = Q_{RT})$. The predicted packet arrival time $\hat{T}_{next}^{(i)}$ and the predicted packet size $\hat{S}_{next}^{(i)}$ of Equation (16) may be obtained from Equations (7) and (8) as previously described.

To generate predicted time $\hat{t}_{finish,2}^{(i)}$ in accordance with Equation (15), the predicted number of cells at the real-time buffer 50 $Q_{RT}^{pred}(W^{(i)})$ may be obtained from Equation (17):

$$Q_{RT}^{pred}(W^{(i)}) \approx \max\left\{0, Q_{RT} + \sum_{\substack{j=1 \\ j \neq i}}^{n_v}\left\lceil \frac{\hat{S}_{tot}^{(j)}(W^{(i)}) + 24}{184}\right\rceil - \frac{R_{encap} W^{(i)}}{(204)(8)}\right\} \tag{17}$$

15

where $\hat{S}_{tot}^{(j)}(W^{(i)})$ is described as the number of additional bytes that could accumulate at queue j as defined by Equations (18) through (22):

$$j = \sum_{k=1}^{r^*} \hat{S}_j(m_{last}^{(j)};k) \tag{18}$$

5   where the recursion variable r^* is the largest integer that satisfies Equation (19):

$$\sum_{k=1}^{r^*} \hat{T}_j(m_{last}^{(j)};k) \leq W^{(i)} + t_{current} - t_{start}^{(j)}(m_{last}^{(j)}) \tag{19}$$

and where $m_{last}^{(j)}$ is the index of the most recent packet to arrive at the jth queue.

10   According to one embodiment, the recursion variable r^* may be, for example, the largest integer that satisfies Equation (20):

$$\sum_{k=1}^{r(t)} \hat{T}_j(m_{last}^{(j)};k) \leq t \tag{20}$$

15   for any positive real value number t. Thus, using Equation (7) to obtain predicted time $\hat{T}_j(m_{last}^{(j)};k)$ and manipulating the result into Equation (20), a recursion function $r(t)$ may be obtained as described by Equation (21):

$$r(t) \leq \frac{t}{\mu_{inter}} - \frac{(T_j - \mu_{inter}(j))(\hat{\alpha}_j - (\hat{\alpha}_j)^{r(t)+1})}{\mu_{inter}(j)(1-\hat{\alpha}_j)} \tag{21}$$

20   Finally, once the recursion function $r(t)$ is generated from Equation (21), for example, by using an iterative process, recursion variable r^* may be obtained by substituting time t in Equation (21) with $W^{(i)} + t_{current} - t_{start}^{(j)}(m_{last}^{(j)})$ so that a solution for $\hat{S}_{tot}^{(j)}(W^{(i)})$ may be obtained from Equation (22):

25

$$\hat{S}_{tot}^{(j)} = S_{tot}^{(j)} + (S_j - \mu_{size}(j))\frac{(\hat{\beta}_j - (\hat{\beta}_j)^{r^*+1})}{1-\hat{\beta}_j} + r^* \mu_{size}(j) \tag{22}$$

where $\hat{\beta}_j = \hat{\beta}_j(m_{last}^{(j)})$ and $S_j = S_j(m_{last}^{(j)})$. Thus, using Equation (22) to obtain $\hat{S}_{tot}^{(j)}(W^{(i)})$ for each $j=1,2,...,n_v$, a value for the predicted number of cells at the real-time buffer 50 $Q_{RT}^{pred}(W^{(i)})$ may be obtained from Equation (17) and therefore a delay d_i in accordance with Equation (13) may be obtained.

5   The method proceeds to step 414 to determine if the delay associated with waiting for a next packet is within jitter bounds. The jitter bounds may comprise a predetermined maximum jitter requirement that functions as a delay threshold for evaluating a delay. According to one embodiment, the maximum jitter requirement is, for example, in the range of 25 to 35 milliseconds such as approximately 30 milliseconds. The delay

10   obtained in step 412 is compared with the jitter requirement. A delay is within jitter bounds when the delay is lower than the jitter requirement. If the delay is within jitter bounds, that is, the delay is lower than the jitter requirement, the method proceeds to step 420 to wait and accumulate the next packet expected at the queue. If the delay is not within jitter bounds, that is, the delay is equal to or greater than the jitter requirement, the

15   method proceeds to encapsulation of the accumulated packets in the per-flow queue under step 416 and to subsequently form cells from the encapsulation section at step 418. Cells may be formed in any suitable manner. For example, cells may be formed according to the procedure described with reference to FIGURE 2. After forming the cell, the method terminates.

20   To summarize, if the packets accumulated satisfy the encapsulation efficiency threshold, the predicted jitter is evaluated to determine if the accumulated packets are encapsulated. If the predicted jitter is low, more packets will be accumulated. If the predicted jitter is too high, then the accumulated packets are encapsulated. The method may include more, fewer, or different steps. The steps of the method may be performed in

25   any suitable order. For example, the prediction of the arrival time at step 408 may be performed after the determination at step 410.

           FIGURE 5 illustrates another embodiment of the formation of cells 550 from encapsulation sections 500. According to one embodiment, encapsulation section 500 may comprise a Multi-Protocol Encapsulation (MPE) section. Each encapsulation section

30   500 includes a section header 210, section data 215 and 218, and a section footer 214. Section header 210 may include, for example, digital video broadcasting (DVB) Multi-

17

Protocol Encapsulation (MPE) header data. According to one embodiment, section header 210 may also include, for example, Digital Storage Media Command and Control (DSM-CC) header data. Encapsulation section 500a comprises section data 218 and encapsulation section 500b comprises section data 215, where each section data 218 and

5   215 comprises packets. According to one embodiment, each packet 218 and 215 may comprise an IP packet or datagram. Section footer 214 may include, for example, error correction codes. Datagram sizes may be variable, while section header 210 has, for example, 20 bytes, and section footer 214 has, for example, four bytes.

Encapsulation section 500 may be formed into cells 550. Each cell 550 includes a

10   cell header 220, a payload 540, and a cell footer 230. Cell header 220 may include, for example, MPEG-2 packet header data. Cell footer 230 may include, for example, MPEG-2 packet footer data. Payload 540 may include MPEG-2 packets, which have, according to one embodiment, 184 bytes.

According to one embodiment, one or more encapsulation sections 500 may be

15   formed into one or more sequential cells 550. In the illustrated example, a cell 550a may be loaded with a first portion of encapsulation section 500a, a next cell 550b may be loaded with a next portion of encapsulation section 500a, while another cell 550c may be loaded with a last portion of section 500a. In general, cells may be formed sequentially until an encapsulation section 500 has been formed into one or more cells. In the

20   illustrated example, the last portion 580 of encapsulation section 500a may be loaded into a payload 540c of a cell 550c, and a first portion 585 of a next encapsulation section 500b is loaded into the payload 540c of cell 550c, so that a condition of having additional capacity 280 as described by FIGURE 2 is substantially avoided.

FIGURE 6 illustrates another embodiment of a cell flow 600 resulting from the

25   formation of cells 550 described with reference to FIGURE 5. In general, the cell flow 600 may include packets 212 and 215 ordered sequentially, consecutively, successively, serially, or in any similar manner therewith. According to one embodiment, packets 212 and 215 may include, for example, IP packets or datagrams. In the illustrated example, the cell flow 600 may include cell footer 610 and cell header 620 that may cause a delay 350.

30   The delay may result substantially from cell footer 610 and cell header 620 or may include any other suitable content. According to one embodiment, a cell flow 600 may be obtained to improve a delay or jitter. In the illustrated example, there may be a smaller

18

delay 350 at a cell flow 600 described by FIGURE 6 compared to the delay 350 described by FIGURE 3, where an additional capacity 280 contributes to the content of the delay in the embodiment shown in FIGURE 3. A cell flow 600 as described by FIGURE 6 may therefore improve the delay 350 associated with receiving packets 212 and 215 when, for

5   example, an additional capacity as described by FIGURE 3 has been substantially avoided.

Certain embodiments of the invention may provide one or more technical advantages. A technical advantage of one embodiment may be that the size of an encapsulation section is adjusted in response to a current encapsulation efficiency and a predicted jitter in an effort to control jitter while maintaining efficiency. If the packets

10   accumulated do not satisfy a predicted encapsulation efficiency threshold, the packets are encapsulated to maintain efficiency. Another technical advantage of an example may be that a delay associated with receiving predicted packets may be determined and if the delay satisfies a jitter requirement, the accumulated packets may be encapsulated to control jitter. Another technical advantage of an example may be that encapsulation

15   sections packets may be formed into cells to improve jitter. The encapsulation sections may be formed into cells so that additional capacity in the cells is avoided.

Although an embodiment of the invention and its advantages are described in detail, a person skilled in the art could make various alterations, additions, and omissions without departing from the spirit and scope of the present invention as defined by the

20   appended claims.

19

WHAT IS CLAIMED IS:

1.   A method for encapsulating packets, comprising:

receiving a plurality of packets at an encapsulator comprising a queue;

repeating the following for each packet of a subset of the plurality of packets until

5   the subset of the plurality of packets is encapsulated:

accumulating a packet at the queue;

determining a current encapsulation efficiency associated with one or more

packets accumulated at the queue;

predicting a next encapsulation efficiency associated with the one or more

10   packets accumulated at the queue and with a predicted next packet;

encapsulating the one or more packets accumulated at the queue if the

current encapsulation efficiency satisfies an encapsulation efficiency threshold and if the

current encapsulation efficiency is greater than the next encapsulation efficiency; and

generating a section from the encapsulated packets.

15

2.   The method of claim 1, wherein repeating the following for each packet of

a subset of the plurality of packets until the subset of the plurality of packets is

encapsulated further comprises:

determining a section size associated with the one or more packets accumulated at

20   the queue; and

encapsulating the one or more packets accumulated at the queue if the section size

satisfies a maximum section size.

3.   The method of claim 1, wherein repeating the following for each packet of

25   a subset of the plurality of packets until the subset of the plurality of packets is

encapsulated further comprises:

predicting a delay associated with the packet and a next packet; and

encapsulating the one or more packets accumulated at the queue if the delay does

not satisfy a jitter requirement.

30

4.   The method of claim 1, wherein the encapsulation efficiency threshold comprises a minimum encapsulation efficiency.

5.   The method of claim 1, wherein the current encapsulation efficiency comprises a ratio of a number of payload bytes associated with a predicted section comprising the one or more packets accumulated at the queue to a total number of bytes of the predicted section comprising the one or more packets.

6.   The method of claim 1, wherein predicting a next encapsulation efficiency associated with the one or more packets accumulated at the queue and with a predicted next packet comprises:
        predicting a next packet size; and
        calculating the next encapsulation efficiency in accordance with the next packet size.

7.   The method of claim 1, wherein predicting a next encapsulation efficiency associated with the one or more packets accumulated at the queue and with a predicted next packet comprises:
        predicting a next packet size according to an autoregressive model; and
        calculating the next encapsulation efficiency in accordance with the next packet size.

8.   The method of claim 1, further comprising forming a cell comprising at least a portion of the section.

9.   The method of claim 1, further comprising:
        forming a first cell comprising at least a portion of the section, the first cell comprising additional payload capacity; and
        forming a second cell comprising a next section.

10.   The method of claim 1, further comprising forming a cell comprising a first portion of the section and a second portion of a next section.

11.   The method of claim 3, wherein predicting a delay associated with the
5   packet and a next packet comprises:

recording a first arrival time and a first departure time of the packet;

recording a second arrival time of a next packet;

predicting a second departure time of the next packet;

generating a first difference between the first departure time and the second
10   departure time;

generating a second difference between the first arrival time and the second arrival time; and

predicting the delay in accordance with the first difference and the second difference.

15

12.   The method of claim 3, wherein the jitter requirement comprises a maximum jitter requirement.

13. A system for encapsulating packets, comprising:

a queue operable to receive a plurality of packets; and

a processor coupled to the queue and operable to:

5   repeat the following for each packet of a subset of the plurality of packets

until the subset of the plurality of packets is encapsulated:

accumulating a packet at the queue;

determining a current encapsulation efficiency associated with one

or more packets accumulated at the queue;

10   predicting a next encapsulation efficiency associated with the one or

more packets accumulated at the queue and with a predicted next packet;

encapsulating the one or more packets accumulated at the queue if

the current encapsulation efficiency satisfies an encapsulation efficiency threshold and if

the current encapsulation efficiency is greater than the next encapsulation efficiency; and

15   generate a section from the encapsulated packets.

14. The system of claim 13, further comprising:

at least one buffer operable to receive a plurality of cells from the queue, the at

least one buffer comprising at least one real-time buffer associated with a first priority;

20   and

a scheduler operable to control a flow of the plurality of cells from the at least one

buffer, the scheduler being operable to grant priority to the plurality of cells received from

the at least one real-time buffer.

25   15. The system of claim 13, wherein the processor operates to:

determine a section size associated with the one or more packets accumulated at

the queue; and

encapsulate the one or more packets accumulated at the queue if the section size

satisfies a maximum section size.

16.   The system of claim 13, wherein the processor operates to:

predict a delay associated with the packet and a next packet; and

encapsulate the one or more packets accumulated at the queue if the delay does not satisfy a jitter requirement.

5

17.   The system of claim 13, wherein the encapsulation efficiency threshold comprises a minimum encapsulation efficiency.

18.   The system of claim 13, wherein the current encapsulation efficiency

10   comprises a ratio of a number of payload bytes associated with a predicted section comprising the one or more packets accumulated at the queue to a total number of bytes of the predicted section comprising the one or more packets.

19.   The system of claim 13, wherein the processor operates to:

15   predict a next packet size; and

calculate the next encapsulation efficiency in accordance with the next packet size.

20.   The system of claim 13, wherein the processor operates to:

predict a next packet size according to an autoregressive model; and

20   calculate the next encapsulation efficiency in accordance with the next packet size.

21.   The system of claim 13, wherein the processor operates to form a cell comprising at least a portion of the section.

25   22.   The system of claim 13, wherein the processor operates to:

form a first cell comprising at least a portion of the section, the first cell comprising additional payload capacity; and

form a second cell comprising a next section.

30   23.   The system of claim 13, wherein the processor operates to form a cell comprising a first portion of the section and a second portion of a next section.

24.   The system of claim 15, wherein the processor operates to:

record a first arrival time and a first departure time of the packet;

record a second arrival time of a next packet;

predict a second departure time of the next packet;

generate a first difference between the first departure time and the second departure time;

generate a second difference between the first arrival time and the second arrival time; and

predict the delay in accordance with the first difference and the second difference.

25.   The system of claim 16, wherein the jitter requirement comprises a maximum jitter requirement.

26. A method for encapsulating packets comprising:

receiving a plurality of packets at an encapsulator comprising a queue;

repeating the following for each packet of a subset of the plurality of packets until the subset of the plurality of packets is encapsulated:

5   accumulating a packet at the queue;

predicting a delay associated with the packet and a predicted next packet; and

encapsulating the one or more packets accumulated at the queue if the delay does not satisfy a jitter requirement; and

10   generating a section from the encapsulated packets.

27. The method of claim 26, wherein predicting a delay associated with the packet and a next packet comprises:

recording a first arrival time and a first departure time of the packet;

15   recording a second arrival time of the next packet;

predicting a second departure time of the next packet;

generating a first difference between the first departure time and the second departure time;

generating a second difference between the first arrival time and the second arrival

20 time; and

predicting the delay in accordance with the first difference and the second difference.

28. The method of claim 26, wherein predicting a delay associated with the

25 packet and a next packet comprises:

predicting a next packet arrival time according to an autoregressive model; and

calculating the delay in accordance with the next packet arrival time.

29. The method of claim 26, wherein the jitter requirement comprises a

30 maximum jitter requirement.

26

30.   A system for encapsulating packets, comprising:

a means for receiving a plurality of packets; and

a means for repeating the following for each packet of a subset of the plurality of packets until the subset of the plurality of packets is encapsulated:

accumulating a packet at the queue;

determining a current encapsulation efficiency associated with one or more packets accumulated at the queue;

predicting a next encapsulation efficiency associated with the one or more packets accumulated at the queue and with a predicted next packet;

encapsulating the one or more packets accumulated at the queue if the current encapsulation efficiency satisfies an encapsulation efficiency threshold and if the current encapsulation efficiency is greater than the next encapsulation efficiency; and

means for generating a section from the encapsulated packets.

31.   A method for encapsulating packets, comprising:

receiving a plurality of packets at an encapsulator comprising a queue;

repeating the following for each packet of a subset of the plurality of packets until the subset of the plurality of packets is encapsulated:

5   accumulating a packet at the queue;

determining a current encapsulation efficiency associated with one or more packets accumulated at the queue, the current encapsulation efficiency comprising a ratio of a number of payload bytes associated with a predicted section comprising the one or more packets accumulated at the queue to the total number of bytes of the predicted

10   section comprising the one or more packets;

determining a section size associated with the one or more packets accumulated at the queue;

encapsulating the one or more packets accumulated at the queue if the section size satisfies a maximum section size;

15   predicting a next encapsulation efficiency associated with the one or more packets accumulated at the queue and with a predicted next packet;

encapsulating the one or more packets accumulated at the queue if the current encapsulation efficiency satisfies a minimum encapsulation efficiency and if the current encapsulation efficiency is greater than the next encapsulation efficiency;

20   predicting a delay associated with the packet and a next packet;

encapsulating the one or more packets accumulated at the queue if the delay does not satisfy a maximum jitter requirement;

generating a section from the encapsulated packets; and

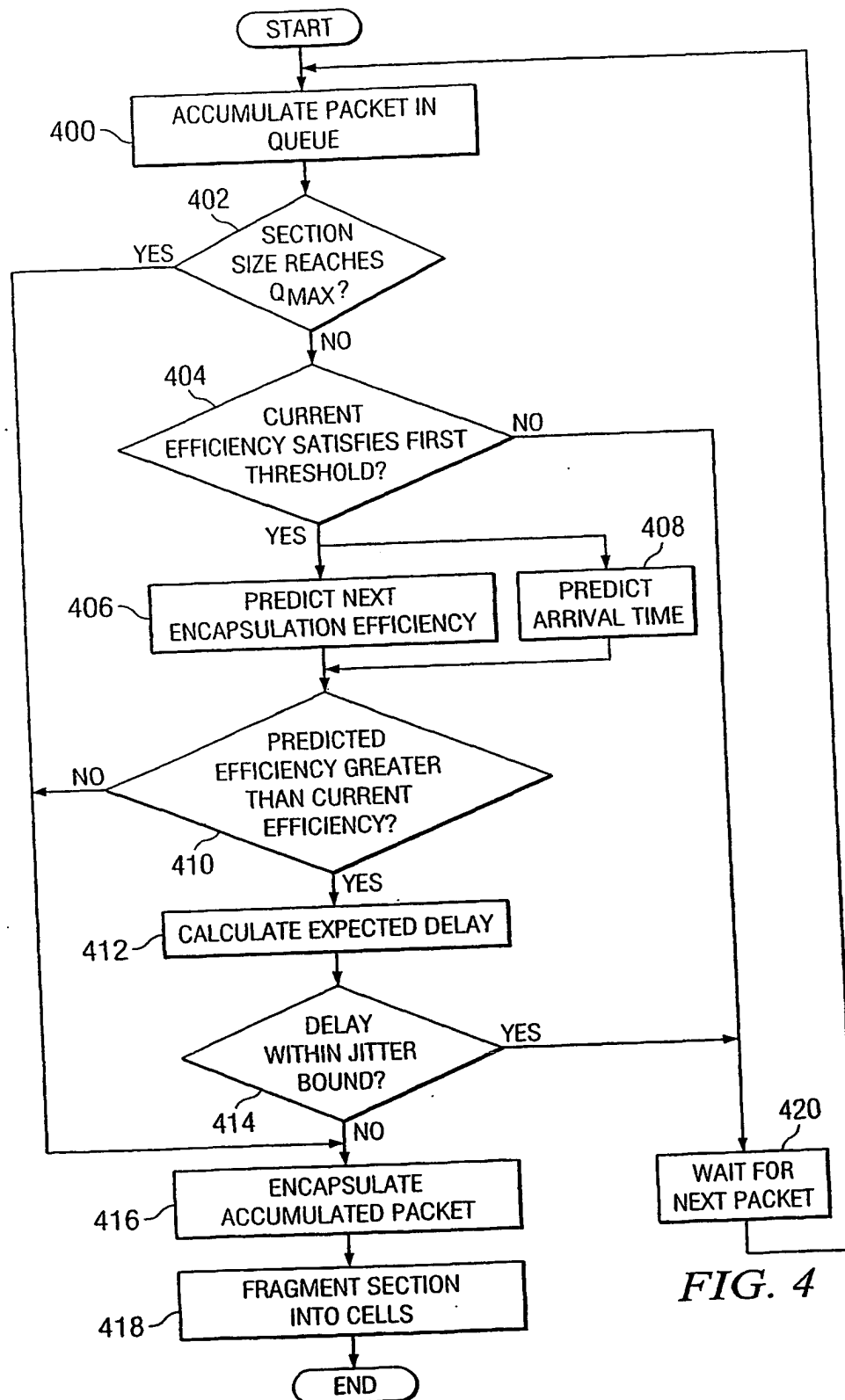forming a cell comprising at least a portion of the section.

1/3

FIG. 1

FIG. 2

FIG. 3

FIG. 5

FIG. 6
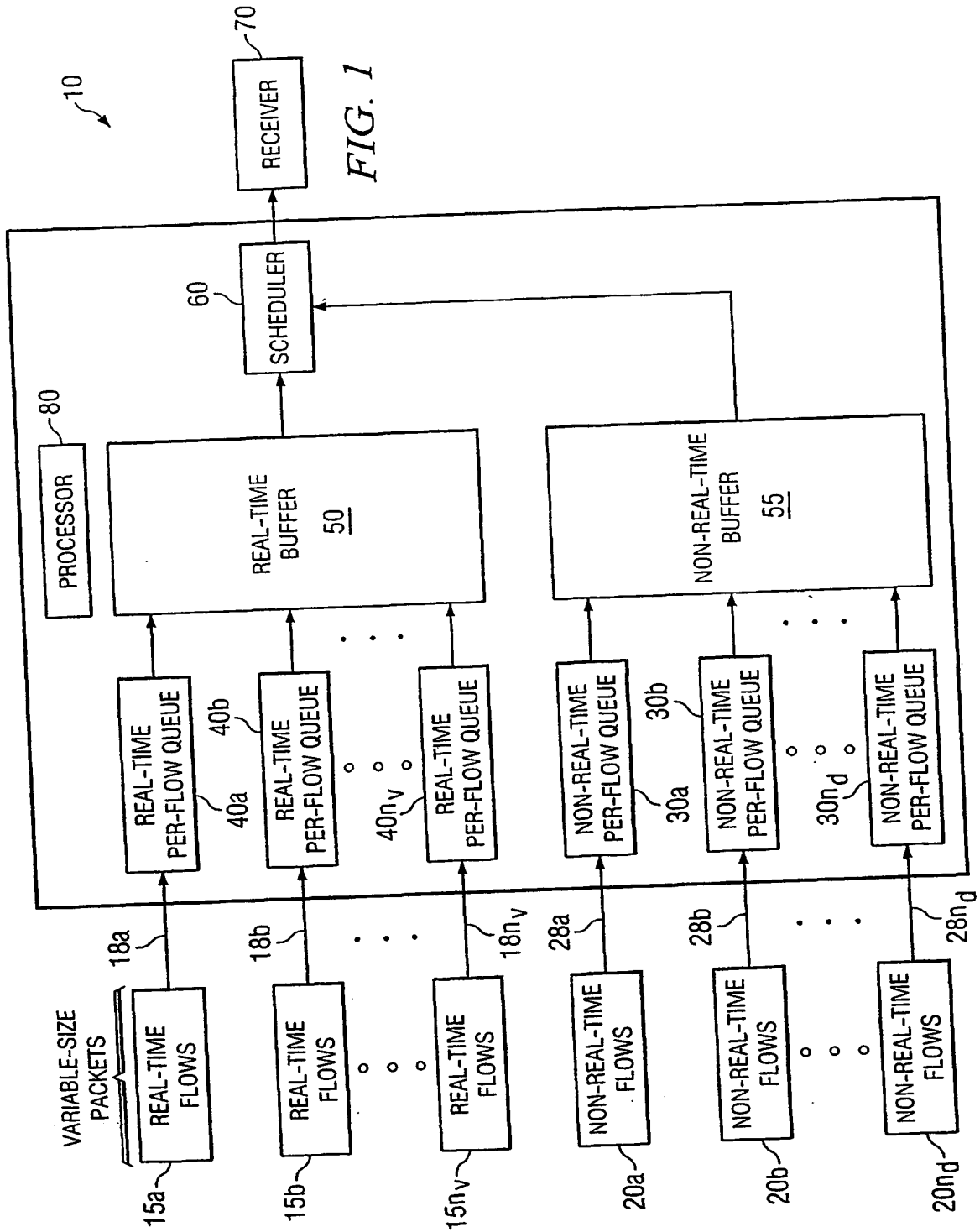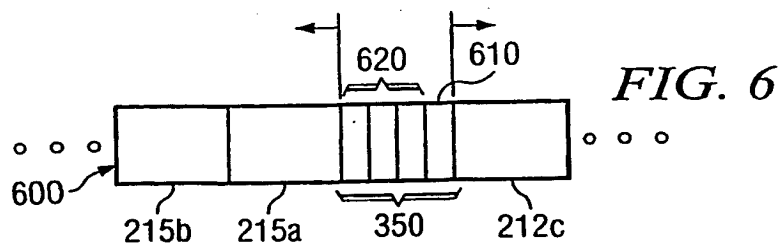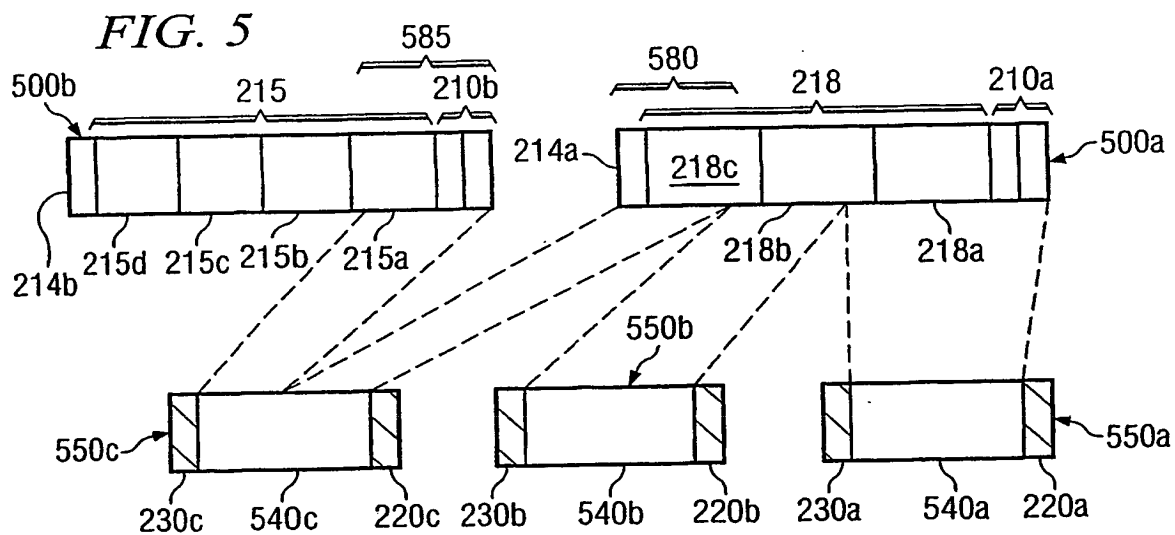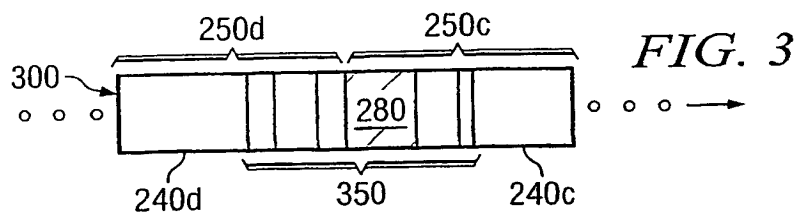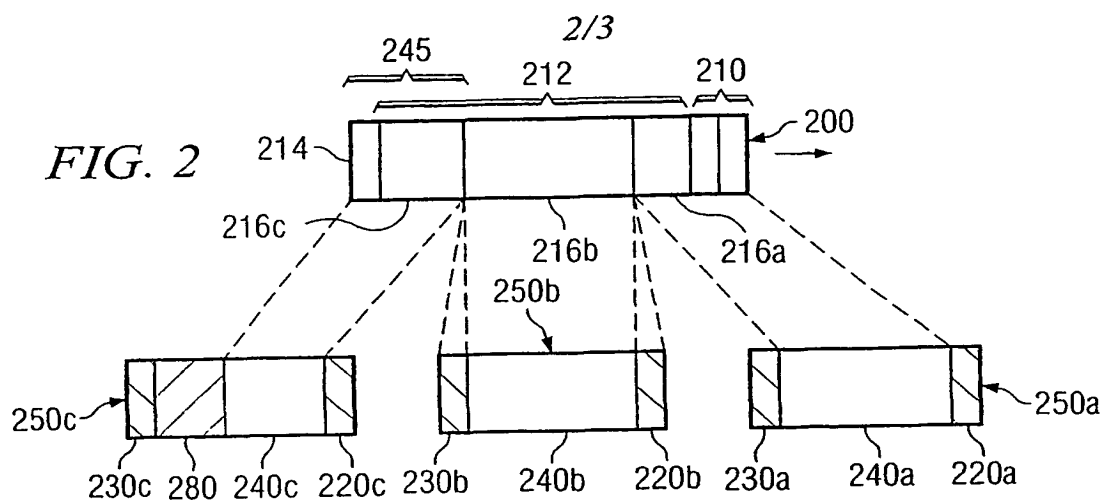
FIG. 4

INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER

IPC 7   H04L12/46

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 7   H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data, PAJ, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5 537 408 A (LEICHTY PHILLIP L ET AL) 16 July 1996 (1996-07-16) column 1, line 45 -column 2, line 60 column 5, line 65 -column 7, line 30 column 8, line 10 - line 30 column 10, line 15 - line 35 column 11, line 20 - line 40 column 12, line 10 - line 40 ---	1-5,8, 12-18, 21,25, 26,29-31
A	EP 1 178 635 A (CIT ALCATEL) 6 February 2002 (2002-02-06) paragraph '0011! paragraph '0015! paragraph '0048! - paragraph '0051! ---	1-4,11, 13-17, 24,26, 27,30,31

-/--

[X] Further documents are listed in the continuation of box C.   [X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
7 April 2004	16/04/2004

Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Authorized officer Litton, R

Form PCT/ISA/210 (second sheet) (January 2004)

International Application No

PCT/US 03/30957

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 98/47293 A (BAHLENBERG GUNNAR ;LJUNGGREN LIS MARIE (SE); TONVALL HARRY (SE); D) 22 October 1998 (1998-10-22) page 12, paragraph 3 -page 13, paragraph 1 -----	1,13,26, 30,31

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
US 5537408	A	16-07-1996	US	5533021 A	02-07-1996
			EP	0725506 A2	07-08-1996
			JP	2898914 B2	02-06-1999
			JP	8321836 A	03-12-1996
			JP	3130294 B2	31-01-2001
			JP	11261600 A	24-09-1999
			US	5652749 A	29-07-1997
EP 1178635	A	06-02-2002	EP	1178635 A1	06-02-2002
			US	2002018443 A1	14-02-2002
WO 9847293	A	22-10-1998	SE	519917 C2	22-04-2003
			EE	9900491 A	15-06-2000
			EP	0986915 A1	22-03-2000
			NO	994939 A	16-12-1999
			SE	9701415 A	17-10-1998
			WO	9847293 A1	22-10-1998